1-1-2001

Differentially secure multicasting

Stephanie Lee Holeman
Iowa State University

Differentially secure multicasting

by

Stephanie Lee Holeman

A thesis submitted to the graduate faculty
in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE

Major: Computer Engineering
Major Professor: Manimaran Govindarasu

Iowa State University
Ames, Iowa
2001

Graduate College

Iowa State University

This is to certify that the Master's thesis of

Stephanie Lee Holeman

has met the thesis requirements of Iowa State University

Signatures have been redacted for privacy

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

In this age of information, the efficient use of electronic communications is essential. As technology advances and becomes more complex, it is imperative that groups be able to discuss ideas and disseminate information among members effectively. Multicast groups are established to facilitate these information transactions. Since the members of these groups may be spread across the globe, the communications must be secure as well as efficient. Secure multicasting is an active area of research today.

Though the areas of secure multicast group architecture, key distribution, and sender authentication are under scrutiny, one topic that has not been explored is how to integrate these with multilevel security. Multilevel security is the ability to distinguish subjects according to classification levels, which determines to what degree they can access confidential objects. In the case of groups, this means that some members can exchange messages at a higher sensitivity level than others. The Bell-La Padula model outlines the rules of these multilevel accesses. In multicast groups that employ multilevel security, some of these rules are not desirable so a modified set of rules was developed and is termed differential security.

This thesis proposes three possible methods in which to set up a differentially secure multicast group: a naïve approach, a multiple tree differential security (*DiffSec*) approach, and a single DiffSec tree approach. In order to evaluate the performances (in terms of the number of links used per packet transmitted) of these approaches, extensive simulation experiments were conducted by varying the network connectivity and group size for both uniform and nonuniform membership distribution across security levels. Our studies show that the multiple tree and single DiffSec tree approaches perform much better than the naïve situation. While the multiple tree approach could be implemented using current technology, this scheme consumes many times more addresses and network resources than the single DiffSec tree approach. From our studies, we conclude that the single DiffSec tree is a viable option for supporting multilevel security as it maximizes the resource utilization and is also scalable.

1. INTRODUCTION TO SECURITY

It is readily apparent that electronic communications are vital in today's world. These communications are at a higher volume than they have ever been and the amount will only continue to increase for the foreseeable future. Since the beginning of computer networks, when they were used mostly for exchanges of academic ideas or commercial processing, the ways in which they are used have constantly expanded. No longer limited to a few academic or commercial applications, electronic communications now include conversations between individuals, commerce between businesses and consumers, financial transactions for investing or managing assets, inquiring or exchanging information regarding health issues, and a myriad of other services.

As additional uses developed and more entities became interested in these communications, the number and size of the networks has had to expand. To keep transactions cost-efficient, networks have been coupled to facilitate exchanges and allow a broader reach of electronic communications. Now communications flow between as well as within networks. Though this inter-network exchange is beneficial, it also means that control over the flow of information is lost. In order to regain some control over who can intercept, read, or change messages while they are in transit, the concepts of computing security were developed, which is part of the broader field of information assurance.

Computing security makes use of cryptology, which plays an important role in this thesis. To secure data in a broadcast network environment, it is encrypted. Groups of hosts maintain security through the use of encryption algorithms and shared cryptographic keys.

1.1 Goals of Security

The goals of computing security are to assure information confidentiality, integrity, and availability [Pf97]. The following definitions will focus on assurance as it relates to communication, though these goals relate to securing all computing resources. Confidentiality, also known as secrecy or privacy, relates to the ability to ensure that only authorized entities are able to read or know the contents of a message. Assurance of integrity means a guarantee that the communication was modified only by an authorized party and

only in a prescribed way. To assure availability is to make certain that authorized parties are not prevented from receiving or sending communications when they have legitimate access, or in other words, that they are not denied service.

These goals address the methods that are used to attack information. The threats to secure communications include interception, modification, fabrication, and interruption [Pf97]. The first three involve reading, changing, or inventing the contents of a message by an unauthorized party. The last involves blocking the communication from reaching the intended recipient or corrupting the message so that it can no longer be read. Preventing interruption of network communication is a topic in and of itself and beyond the scope of this thesis. It is not always possible to prevent interception, modification, or fabrication, but by using the mathematical functions of encryption, it is possible to make the contents of an intercepted message meaningless to an unauthorized party and to alert an authorized recipient to modified or fabricated messages.

1.2 Encryption

Encryption is the method of taking readable text (*plaintext*) and processing it through an algorithm to create a text whose meaning is non-obvious (*ciphertext*). The algorithm that is used can be secret or well known. Since it is impractical for everyone to create their own cryptographic algorithm, the secrecy in a cryptosystem usually relies upon some further input (a *key*) in order to create the ciphertext. Depending on the intended use of the encryption, the key will be secret (*private key*) or well known (*public key*).

Decryption is the method of translating ciphertext back to the original plaintext. The decryption algorithm also needs a key input in order to calculate the correct result. This may be the same key that was used when encrypting or it may be a related key, depending upon the cryptosystem being used. The two main categories of encryption schemes and their uses are explained below.

1.2.1 Symmetric Cryptography

When the key used for decryption is the same as, or easily deduced from, the encryption key, this arrangement is known as a symmetric cryptosystem. Other terms include secret-key or private-key cryptosystem. (Since the term private-key is used in other

cryptosystems, this thesis will use the term secret-key when referring to symmetric cryptography.) It is necessary to keep the key a secret between the sender (*encrypter*) and the receiver (*decrypter*) in order to maintain the security of the information. Implementations include algorithms such as the Data Encryption Standard (DES) and the International Data Encryption Algorithm (IDEA). A full delineation of these algorithms is beyond the scope of this thesis but the following references can be consulted for more information: [NBS77], [Sc94].

One of the challenges with symmetric cryptosystems is the question of how to distribute the key. It cannot simply be transmitted over the network because opponents could simply observe the value as it is sent. One solution is to use a secure channel to pre-distribute keys to the parties that need it. Some methods to accomplish this include the Blom key [Bl85] and the Diffie-Hellman key [DH76] pre-distribution schemes. Since a secure channel may not always be available, there are on-line methods, such as Kerberos [KN93], for key distribution. The majority of these methods, both secure channel and on-line, make use of an intermediary. This intermediary is a trusted authority that can authenticate itself to each party and communicate securely with them. This authentication is in the form of a computed signature, which relies on the use of public information. Signature schemes generally involve the use of asymmetric cryptography.

1.2.2 Asymmetric Cryptography

An asymmetric, or public-key, cryptosystem is one in which the decryption key is not easily computed from the encryption key [DH76]. Thus, the encryption key can be made public knowledge so that anyone may use it to encrypt a message. However, only the receiver knows the decryption key (which is also known as a private key) that is necessary decrypt the ciphertext. The decryption and encryption key are mathematically related

The calculation of these key pairs involves one-way functions. A one-way function is one that is easy to compute but hard to invert. More formally,

> A function f: X → Y is called one-way if for any x that is an element of X it is easy to compute f(x), but given any y that is an element of Y it is computationally infeasible to find an x such that f(x) = y. [Be00]

Though one-way functions are important in constructing asymmetric cryptosystems, there are currently no functions that have been mathematically proven to be one-way; yet, there is considerable evidence that they do exist [St95].

Because public-key cryptography uses exponentiation, it tends to be much slower (i.e. takes more computation time) than symmetric cryptographic methods. Typically, public-key cryptography is used to exchange a randomly chosen secret-key that will then be used for the communication session and then messages are encrypted using a symmetric algorithm. The most commonly used implementation of public-key cryptography is the Rivest-Shamir-Adelman (RSA) cryptosystem [RSA78].

1.3 How Encryption Secures

Though they use keys differently, both symmetric and asymmetric cryptosystems protect information through the employment of encryption. Encryption can ensure security in the event of interception. Before a message is sent, it can be encrypted using either the shared secret-key or the intended receiver's public-key, depending on which cryptosystem is in use. Only a receiver with knowledge of the secret-key or the private-key that corresponds to the public-key used for encryption can decrypt the message. As mentioned above, if an unauthorized agent sees the communication, it will not matter because the contents will be unintelligible.

Encryption also allows detection of message modification. Assuming that the adversary does not know the encryption key, if a part of an encrypted message is changed or deleted by the adversary, then the receiver will not be able to decrypt part or all of the message. The part that was modified will look garbled and will at least register as a transmission error if not actually be detected as an attack on the communication.

Finally, encryption protects against the fabrication of messages. An antagonist who does not have the secret-key or the private key, cannot encrypt a message in such a way that the receiver would be fooled into thinking the message originated from an authorized sender. Attempts to decrypt the antagonist's message using the correct key, which does not correspond to the antagonist's encrypting key, will result in an unintelligible message.

In addition to encrypting messages, public-key cryptography is used for signing messages. Whether the message itself is encrypted or not, the signature can be used to authenticate the sender and verify the contents. The signature is the result of creating a hash (*message digest*) of the message and then encrypting it using the sender's signing key (private-key) [St99]. The hash function can be well known. The receiver decrypts the signature using the verification key (public-key of the sender), computes the hash of the message, and compares the result with the received hash. If the hashes match, the message retained its integrity during transmission. If the hashes do not match, there is a reason to suspect alteration or fabrication of the message or signature.

2. SECURE MULTICASTING

With the increased reliance on electronic communications, as mentioned in chapter 1, as well as with the ever-compounding complexity of technology, it is increasingly rare for an individual to work alone. Best practice in business and academic curricula is a focus on teams of people that work on projects. In order to accomplish their goals, these teams need efficient ways to communicate amongst the group. There are many examples of such group-oriented communications where information needs to be disseminated among a number of people: bodies of legislature, management hierarchy of a company/university, conferences and/or classroom situations.

These group-oriented communications can be accomplished by a variety of methods: multiple unicast, broadcast, or multicast. Multiple unicast entails sending a point-to-point (unicast) message to each group member. For n senders and r receivers, the number of messages sent is $n*r$. Although this ensures that every group member receives the message, the obvious drawback is the lack of scalability and the waste of bandwidth that occurs when a message is sent multiple times along links that can potentially be shared among multiple group members. Broadcasting is a way to send a message to all parties, interested or not. For n senders, n messages are sent and every network entity receives a message, even if the entity is not interested in the message. In the case of broadcasting, the benefit is that all group members receive the communication with little or no routing. However, it is not efficient in terms of network usage since the message is sent to non-group members as well. Obviously, there is a scalability problem. Multicast was developed as a way to send a message to more than one member while conserving resources by propagating the message only as far as it needs to go to reach each group member and only once along each path. For n senders, n messages are sent and routers duplicate packets as necessary to forward along a path to an interested receiver.

2.1 Multicasting Defined

A multicast group was developed from the concept of the host group model is defined as which "a host group is a set of network entities sharing a common identifying multicast

address, all receiving any data packets addressed to this multicast address by senders (sources) that may or may not be members of the same group and have no knowledge of the groups' membership"[CD85]. The conservation in resources (versus the aforementioned multiple unicast schemes) stems from the fact that routers between sources and receivers only send out one copy of an incoming multicast packet per link [D88]. A multiple unicast scheme has routers sending a copy for each of the receivers that can be accessed using that link. Another advantage is that the source only has to know one multicast address instead of multiple unicast addresses, which is beneficial in cases where group membership changes frequently, making it cumbersome for a source to keep all addresses updated. Although the host group definition allows for non-members to send to the multicast address, secure multicasting designates that all senders must be authorized, whether or not they are also members of the group.

Multicasting is useful in many applications [BA99] [Ra00] [SGLA99] [SM00]. Figure 1 shows the multiple models that categorize these applications. One instance is for one-to-many (1toM) communication, also known as single source communication. This occurs when one source continuously sends out information to many members for applications such as a stock quote feed, a video or audio broadcast, or announcements. A second category is that of many-to-many (MtoM), when members of the group may be senders as well as receivers therefore requiring the ability to perform two-way multicast.

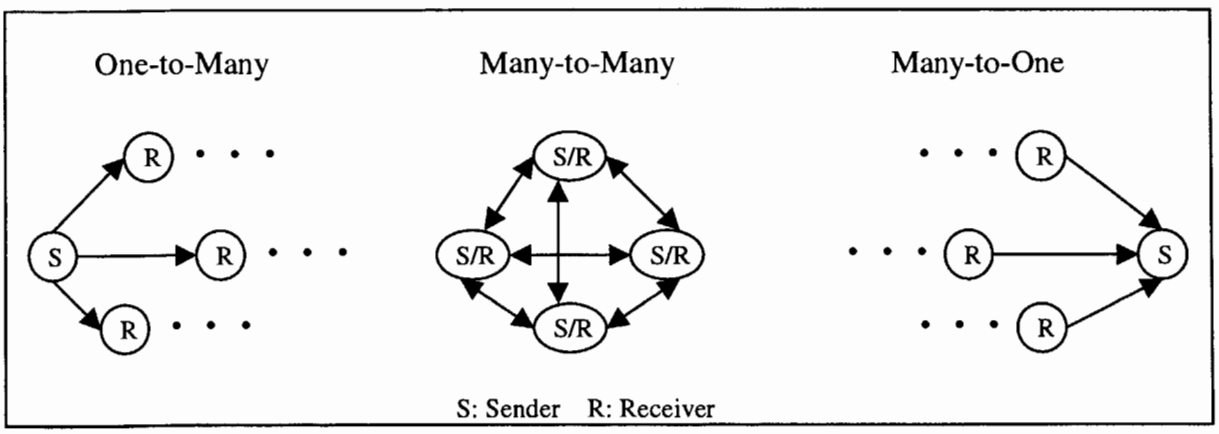

Figure 1. Multicast models [HCD00]

Uses for MtoM communication include collaboration among colleagues, voice, video, or text conferencing, and multi-player games. The third use of multicasting is many-to-one (Mto1) communications. This scenario involves numerous receivers sending data to one source and might be used for voting, auctions, or other data collection applications.

In order to characterize a multicast application, the following group parameters must be known:

- **pervasive *vs.* sparse** -- Whether the group is *pervasive*, having members on most links or subnets or the network, or *sparse*, having members on a small number of widely separated links.
- **open *vs.* closed** -- Whether a group is *open*, allowing senders to be non-members, or *closed*, requiring senders to be group members.
- **permanent *vs.* transient** -- Whether a group is *permanent*, existing eternally or at least for significant lengths of time, or *transient*, existing only for a short duration.
- **static *vs.* dynamic** -- Whether the membership of the group is *static*, remaining relatively constant over time, or *dynamic*, allowing members to join and leave the group.

2.2 Multicast Routing

There are two broad approaches for constructing multicast trees [DM78] [BFC93]: *source-based* tree and *shared* tree (also known as *core-based* tree). In the source-based tree approach, a separate tree is constructed for each sender and rooted at that sender node. Though this approach can offer better performance (e.g. delay) guarantees, it is not scalable since multiple trees must be created for a many-to-many communication situation. The core-based tree approach is a routing scheme considered ideal for many-to-many communications. This approach identifies a core node for the multicast group and constructs a distribution tree rooted at this core and spanning all the group members. Multicast packets from senders who are not members of the group are forwarded toward the core until they reach a node of the distribution tree. From that point, the packet is forwarded to group members as dictated by the nature of the distribution tree. The core-based tree approach accommodates both closed and open groups and uses a single shared tree for the entire group. The advantages of a core-

based tree are its ability to adapt to dynamic multicast groups, its suitability for sparsely distributed receivers, and its scalability to handle large numbers of senders.

Not only are there numerous routing protocols, as seen in [Ra00] [SGLA99] [SM00], but there are multiple ways to manage a group. The group management issue only compounds when group communications need to be secure. Securing multicast communications involves distributing cryptographic keys to the members so that each can encrypt and decrypt messages as appropriate. To maintain the security of encrypted packets, these keys must be recalculated and redistributed at designated times or upon certain events, such as a new member being added or removed from the group. Not only does the manager need to be aware of membership changes but the manager must propagate the consequences of these membership changes to the rest of the group.

In order to understand and evaluate secure multicast schemes, metrics must be defined to serve as a gauge for their effectiveness in solving the multicast problem. These metrics and the subjects of secure multicast architectures, group key management, and packet source authentication [MRR99] are outlined below.

2.3 Metrics

Numerous criteria are used to analyze secure multicast solutions [MRR99] [CGIMNP99]. These criteria are categorized into group membership management, network resource consumption, receiver resource requirements, sender resource requirements, and dependency upon particular standards. Each of these categories is elucidated below.

Group membership management criteria address the concerns of who is and is not part of the group, what the group looks like, and what happens if the group changes. Questions to consider when evaluating a solution's membership management capabilities are shown in Table 1.

Another category of criteria is under the heading of network resource consumption. These are concerned with the load on the network for various stages of the multicast communication process. When analyzing bandwidth consumption of a solution, it is important to note how many messages must be transmitted each time a member joins or leaves and how large the control messages (those for managing the group) are in relation to

Table 1. Evaluation of multicast membership management capabilities

Metric	Questions
Scalability	• Can the solution handle very large, widely distributed groups?
fail/restore gracefully	• When the network or system experiences errors, do they cause an entire failure or is there a graceful degradation of service? • If part or all of a group is isolated from communications due to network or system failures, can those members be restored without needing to reinitialize the entire group?
join/leave secrecy	• When a member joins, can the new member read any past group messages? • When a member leaves, are future group communications protected from being read by the member who has exited?
group dynamics	• Is the solution able to handle groups whose membership frequently changes by parties joining and leaving or is best for a static group? • Can the solution cope with peak situations, when multitudes of members wish to join (or leave) simultaneously? • Is the solution valuable for short-lived groups or long-term groups?
vulnerability to collusion	• Is there a way for a number of non or former members to affect the security of messages? • How long does message secrecy last (is it ephemeral or long-term)? • Does the solution explicitly provide or allow group or source authentication? • Does the solution allow anonymity or non-repudiation?

the data messages. Also of importance is the volume of communication that can be effectively dealt with and whether the solution can handle bursty traffic.

Receiver and sender resource requirements consider the following:

- How many keys must each member or sender store and how large are these keys?
- What is the processing time involved for the member or sender to, respectively, read or send messages?
- Does the solution allow non-members to send data?
- How many senders are allowed? Must these senders be known in advance of group creation?

The final concern is the dependence upon standards. Does the solution depend upon a particular network protocol or network characteristic (such as stability, in order packet delivery, or reliable transmission)? Does the solution depend upon a particular application?

2.4 Architectures

The above metrics help in evaluation of the variety of multicast solutions. These solutions implement a combination of network architecture and key management schemes. The types of architectures include multilevel and two-level. Two-level architectures are further typified by their use of distributed or centralized top-level [MRR99].

To understand multilevel architectures, we examine the Iolus [M97] infrastructure as an example. An Iolus tree is composed of group security agents (GSAs), which can be further divided into a group security controller (GSC) and group security intermediaries (GSIs). The GSC is the root of the tree and each GSI manages its subgroup's keys and communications, including the routing to other GSIs. The multilevel structure of Iolus stems from the fact that GSIs can exist on multiple levels and GSIs lower on the tree can act as clients for GSIs above them. For an illustration of this structure, see Figure 2 taken from [MRR99]. Since each GSI manages the membership of its own subgroup using its own key and is responsible for translating data from one key to another, membership changes are localized to one subgroup. This means a reduction in the number of key change messages needed for each join and leave. The multitude of GSIs makes this solution robust because

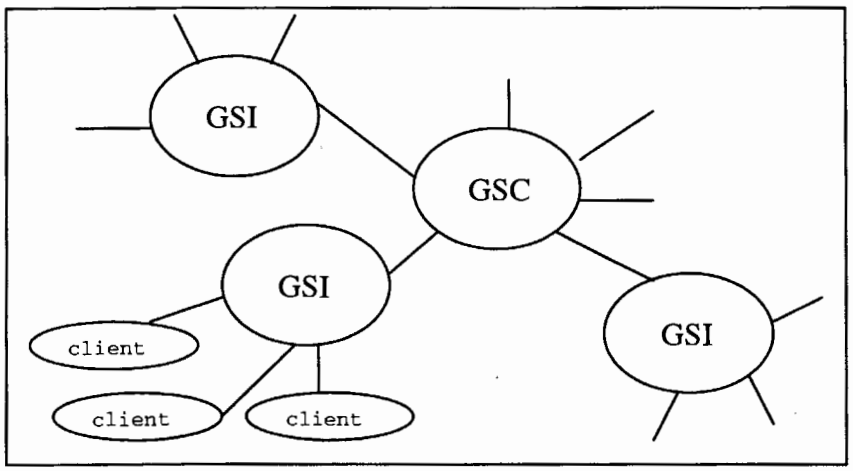

Figure 2. Iolus architecture [M97]

the failure of a single GSI affects only one subgroup. However, there is only one GSC so its failure is fatal to the entire multicast group.

Two-level architectures can have a distributed or centralized top-level. The top-level is an overall control that manages keys for messages between subgroups. This key management can be distributed, as in the Nortel trunk (Figure 3) [HCD00], or centralized, as in the secure reliable multicast (SRM) toolkit [Ch98]. The lower-level can be hierarchical and is divided into subgroups (*leafs* in Nortel structure and *domains* in SRM architecture). As in Iolus, these subgroups localize membership changes and provide robustness against complete group failure. The distributed solution furnishes additional robustness since there is no single controller failure concern as in multilevel and centralized two-level solutions. A centralized two-level solution, while simpler than the multilevel multicast architecture, is less robust and not scalable.

2.5 Key Distribution

As shown in the section on security, encryption is used to secure information and this encryption relies on the use of keys. The problem in multicasting is how to distribute these keys so that each member has the appropriate key at the appropriate time. To ensure perfect join secrecy, so that a new member cannot read previous group messages, and leave secrecy, so that an old member cannot read subsequent messages, the keys must be changed every

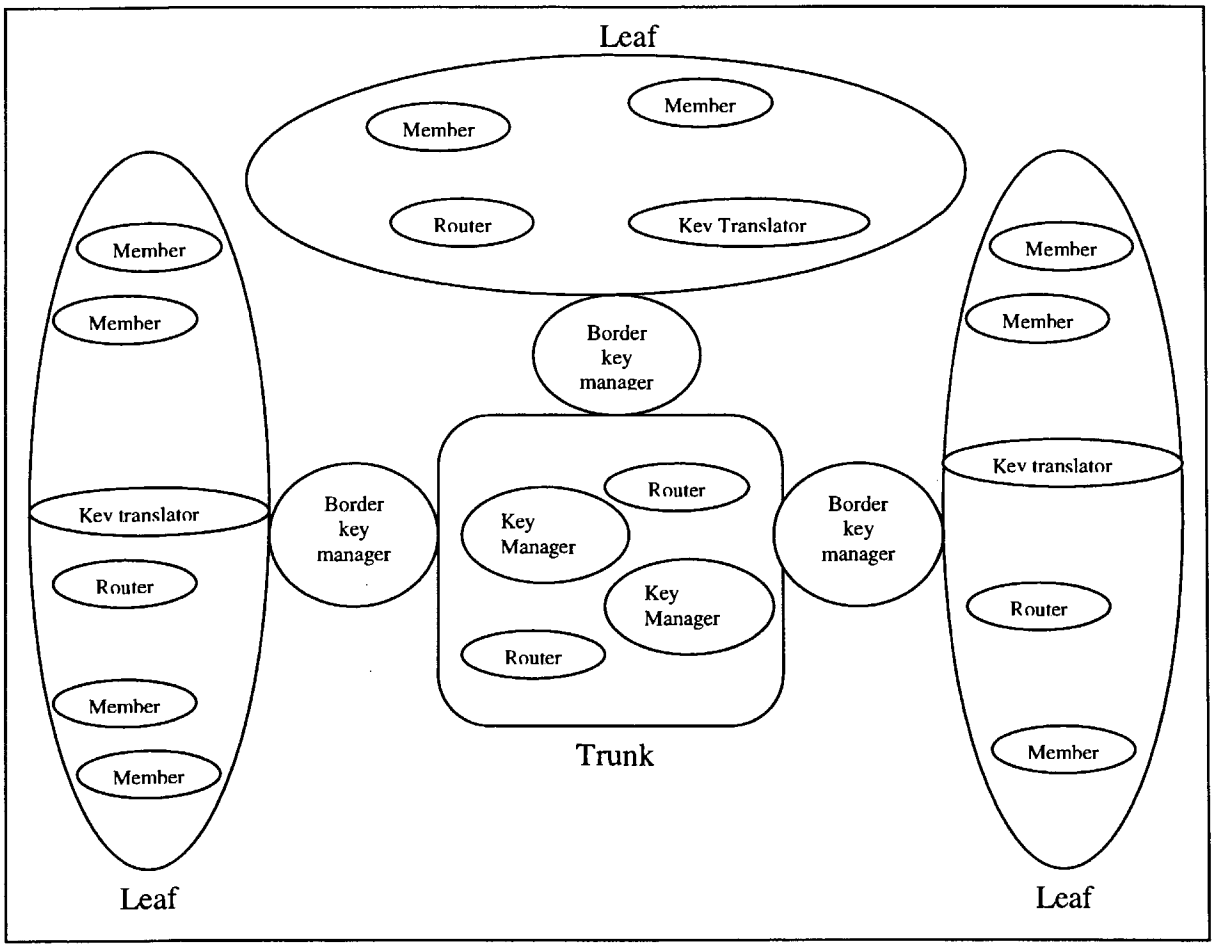

Figure 3. Nortel architecture [HCD00]

time the group membership changes. It is possible to reduce the number of key changes if the rules of perfect join and leave secrecy are relaxed to allow new members to read previous messages or wait for some small number of members to leave before updating the key. Even with the relaxation of rules, it can be difficult to distribute keys efficiently to widely distributed or highly dynamic groups. Other complications include the number of keys the controller and the clients have to store and the processing time involved in calculating new keys. The following strategies attempt to handle key distribution efficiently.

The simplest plan is to merely unicast the group key to each member as needed. The members only have to store the group key and the key used in the unicast with the controller. The controller stores the group key and the keys used to unicast messages to the members.

This arrangement provides for perfect join and leave secrecy and does not require much key storage space. However, it is not efficient for pervasive and dynamic groups since the number of control messages could overtake the number of data messages. It also induces a strain on the controller to process a large number of secure unicast messages.

One of the most widely accepted designs is that of key graphs. Though key graphs can be arranged in star or tree structures, [WGL00] has found the *d*-ary tree to be the most efficient configuration for group key management. This is a hierarchical arrangement of keys such that each member is a leaf on the tree and has as many keys as there are levels in the tree. Each time a join/leave occurs, all of the keys on the path from that member to the root of the tree are changed as well as the siblings of those keys. Though multiple keys are being changed, there are only *d* rekey messages of the *d*-ary tree transmitted. For full detail of d-ary tree key management, refer to [WGL00]. The *d*-ary tree key graph design is highly scalable. The members and controller must store multiple keys but the number is related logarithmically to the size of the group so remains reasonable. Yet, this scheme, just as the simple solution, requires multiple messages to be sent for every join/leave that necessitates rekeying.

One unique approach to key management is that in the SRM toolkit, mentioned above. Each member receives a unique subset of the total set of keys held by the controller. The number of keys stored by members and the controller is determined by the number of members in a domain and is less than the number of keys stored in the d-ary tree scheme. The number of messages sent per leaving member is also less than the tree-based scheme [Ch98]. When a member leaves the group, the new group key is multicast using all the keys not in the subset of the out-going member. Unfortunately, when more than one member leaves, the scheme becomes vulnerable to collusion due to the knowledge gained from the union of the subsets of keys of more than one member.

Other key management solutions such as key agreement amongst peers [STW00] and decentralized key control [PMZ99] attempt to resolve key management issues. All of these have benefits and drawbacks that must be weighed when deciding which to apply. The advantage of hierarchical architecture structures is that they allow the opportunity to employ

different key management schemes for subgroups, thereby using the best-fit solution at each level, depending on the multicast application.

2.6 Multicast Authentication

The subject of multicast authentication is recently one of the most studied. In unicast authentication, the receiver only needs to verify the sender. In the multicast field, there are multiple grades of authentication. Depending upon the authentication scheme, registered receivers are able to determine if a packet came from another registered member, a registered sender, or from a particular registered sender. The first grade, that of authenticating the sender as a registered member, is done by using message authentication codes (MACs) [CGIMNP99]. Receivers must have access to the shared secret key in order to authenticate the sender as a member of the group. Though using MACs is efficient for groups, it is not useful if the receiver wishes a finer granularity of authentication. On the surface it would appear that the authenticating a particular sender is as simple as employing a public-key signing algorithm. The sender signs the message with a private key and the receivers can verify the message using the sender's public key. However, public-key signing algorithms tend to be expensive computationally and consume more bandwidth with longer signatures than other schemes. Efficient methods for verifying registered senders and specifically identifying these senders are under development [CG98] [WL99].

Some solutions involve variations of MACs, such as hashing message authentication codes (HMACS) [KBC97] or multiple keyed MACs [CGIMNP99]. Other solutions make use of streams [GR97] and flows [WL99]. Some authentication methods involve hybrid schemes that use a combination of solutions and offline computations to take advantage of the benefits of other scheme benefits and reducing their drawbacks. Like the key management schemes, there are multiple proposals for multicast authentication no definitive best solution as of yet. The assets and detriments of each plan must be evaluated in context of the desired level of authentication and the type of multicast group. These evaluations are beyond the scope of this thesis.

3. DIFFERENTIAL SECURITY

As the subject of security has developed, several subtopics have been broached. One of these topics focused on the concept of *multilevel security* [Pf97]. This is mostly widely known in its relation to government and, in particular, the military. In this concept, entities not just secure or insecure, they have varying degrees of sensitivity. In the United States military, these degrees are hierarchical in nature and, listed from least secure to most secure, are known as Unclassified, Restricted, Confidential, Secret, and Top Secret. The communication of information is governed by a policy that is need-to-know: sensitive data can only be accessed by parties that need the information in order to perform the duties of their job.

The separating communications by sensitivity levels, or classification*s*, is ubiquitous in the military. Obviously, matters of national security require stringent security regulations. There is a need to provide communication schemes that can handle the concept of classifications. This need exists not only within the military community, but extends to all private sector companies that contract to do business with the military. In reality, even companies that do not contract with the military benefit from such schemes. Though the hierarchy of sensitivity is not as rigid in commercial industry, they do have such classifications as public, proprietary, and internal. They also have corporate structures that involve, possibly from least secure to most secure, employees, middle managers, and managers. So there is the potential for groups other than the military to benefit from schemes that facilitate multilevel communications.

3.1 Defined

Differential security is a term developed for this thesis to designate a partial implementation of multilevel security. In multilevel security where confidentiality is of prime concern, a set of rules directs the flow of information. The Bell and La Padula model formally describes these rules [BL73]. In their confidentiality model, there is a set of subjects S and a set of Objects O. The security class, or *clearance*, of a subject is denoted as

C(s) and the security class, or *classification level*, of an object as C(o). Information can only proceed if it follows the two following properties:

> *Simple Security Property*: A subject s may have read access to an object o only if C(o) <= C(s).
>
> **-Property*: A subject s who has read access to an object o may have write access to an object p only if C(o) <= C(p)

These properties are commonly referred to as *read-down* and *write-up* respectively. This is because a subject may read information from an object that is at its corresponding clearance level or below but may only write information at its clearance level or above. The write-up property does not mean that a subject can alter data at a higher level; it simply means that the subject can make an object, at the subject's security level, more secure by promoting it to a higher classification.

Differential security varies from multilevel in that it does not adhere to the write-up property. While it is desirable to prevent subjects from accidentally or maliciously declassifying information by writing-down, this write-up rule is overly prohibitive since communication with a subject at a lower clearance level is frequently a necessity. An officer must be able to give orders to subordinates and a manager must be able to direct employees. The proposed differential multicasting scheme that follows acknowledges ways to prevent write-down but ultimately rejects this stringent maxim.

Another discrepancy between differential security and the Bell-La Padula model is that of changing security levels. In the Bell-La Padula model, the security classes of subjects and objects are fixed and may not be changed. Differential security recognizes that a subject's clearance level may change, based on appropriate authorization. Military personnel advance in rank, and thus advance in clearance level, and employees are promoted. Differential security allows for this possibility.

Multilevel security not only groups subjects by clearance level, it also designates them by compartment. Compartments, or commonly known as projects, may only contain

information at one sensitivity level or they may span multiple levels. This supports the need-to-know policy by further limiting subjects' realm of knowledge to the information essential to their job performance. The separation of subjects and objects into compartments naturally gives rise to the use of multicasting. A multicast group can be formed for each compartments' subjects (project group). Those subjects, receivers and senders in the group, will only access objects (read/write messages) that pertain to their compartment. By combining multicasting with differential security, the desired means of securing communications, of classifying information and containing it within a compartment, can be achieved.

3.2 Applications Explored

Differentially secure multicasting, as developed in this thesis, fits situations when communications are needed for an entire group as well as for portions of a group. Most one-to-many multicast communications do not need this capability; they usually send all information to all parties. Such is also true of many-to-one communications. Many-to-many multicast uses are more suitable to differential security, as shown below. Due to the overhead of a larger number of rekeying messages whenever a member joins or leaves that has a clearance level greater than the lowest level in the group, this scheme works best when the group is relatively static. However, when group members at the lowest level leave, there is no bandwidth cost difference between this and non-differentially secure multicasting (also referred to in this thesis as regularly secure multicasting). Also related to the number of the key distribution messages, the set-up time for a differentially secure group can be higher than a regularly secure multicast group. Thus, this scheme is not suitable for short-lived groups.

The best-fit application for this scheme is that of a project group. In this many-to-many communication situation, the team members need to be able to synchronize information easily with the group. Managers need to agree with each other as to the direction of the project and send instructions to the project team. The group, once formed, tends to remain relatively static so there is no overload of rekeying traffic for members who leave. There will be some join/leave activity, since different experts are needed at each stage of a project, but if these people are limited to the lowest clearance, there will be no extra traffic when compared to a regular secure multicast group. There is the possibility of promotion

and demotion, which would increase the number of rekeying messages, but these happen infrequently during a typical communication session.

4. PROPOSED APPROACHES

In order to explore the ways to implement differential security in multicasting, the concept itself must be articulated. This chapter first explains differentially secure multicasting and then examines three ways to achieve this concept. Although each approach is functional, they have distinct benefits and drawbacks. A comparison of these approaches will follow their explanations.

4.1 Differentially Secure Multicasting

We begin by defining a notion for a security level. In this thesis, the least secure (lowest clearance) level is referred to as level 1. The highest clearance level is level 4. This could correspond to the military hierarchy with level 1 being Restricted and level 4 as Top Secret. Any material considered Unclassified would not need to be encrypted and can therefore be sent as plaintext and considered to be at level 0. These level designations are not exclusive to military applications. Private industry may define level 0 to be public, level 1 to be proprietary, and level 2 to be internal. There are no limitations in how many levels there are or how each is described; for practical purposes, this thesis will work with groups of four levels of security.

As mentioned in the previous chapter, there is considerable attention paid to who can read what when discussing multilevel security. Since differential security allows a subject to read information that is classified at or below the same level as the subject, each member of a differentially secure multicast group must have all the keys to enable this. A member at level 4 would have keys relating to level 4, level 3, level 2, and level 1 communications. A member at level 1 would only have the level 1 key. Using the notation that C(S) indicates the classification level of an subject (in this case, the encryption key denoted as e_k), then the set of keys held by a members, where S_k is the set of keys, G is the multicast group, and *m* is a member, is as follows:

$$S_k(m)=\{e_k: e_k \in S_k(G) \Lambda C(e_k) <= C(s)\}$$

For each join and leave by member m, all e_k in member *m*'s possession need to be updated. To provide join secrecy, the key(s) for communication are updated whenever a new member

is added. This is easy to do by simply multicasting the new key to existing members and unicasting the new key to the new member. When a member leaves, the task of updating key(s) becomes more involved. Changing all of the group keys that the leaving member possessed means sending secure unicast messages to all members cleared at or below the leaving member's level. However, any member above that level can receive the new keys via secure multicast since the higher-level keys are not affected. One unicast message, containing multiple keys, is sent to all group members who need new keys. (For larger keys, the message will be split into multiple packets.) It is important to note that the level of the key does not indicate its encryption strength. Indeed, all levels may have equally strong encryption. The level simply indicates the communication stream with which the key works. Granted, a level 1 key is typically known by more people so there is a greater chance of collusion with an outsider. There are also more opportunities for attack, since there are a greater number of entities with knowledge of that key.

Requiring the members to hold multiple keys raises the issue of member resources. With a diverse group, some members may be more limited in computation resources than others. Since key sizes are currently in the range of two kilobytes and less, storing as many as four keys does not seem an undue strain on a receiver. If the security hierarchy is considerably deepened, the memory needed to store these keys may become a limiting factor when creating differentially secure groups.

Just as in non-differentially secure multicasting, in order to join the group to receive any keys, the requestor must first authenticate to a group controller. This authority must have the ability to verify the correct clearance level of the joiner and issue all the relevant keys for a member at that level. This is done by way of secure unicast to the joiner.

The authenticating authority could serve as an agent to allow true write-up communications. Thus far, the assumption is that members will only need to write packets at their own level yet read packets at or below their levels. However, implementing the write-up also would allow for the advancement of classification. Since a member does not hold keys for higher levels, any communication that should be advanced in classification needs to be sent to an intermediary who can re-encrypt the information using the higher-level key. The authenticating authority already has possession of all the keys and is presumed to have

greater computing power, since it must be able to handle all requests for changes in group membership, so it would be natural to assign this task as well. However, it is precisely the job of handling authentication that may make the authority a bottleneck if it must also promote information and forward it. The implementer must decide whether the authenticator is an acceptable risk as a bottleneck or if distributing the keys to another third party is an acceptable risk of another potential security leak. One way to avoid using a trusted third party would be to assign read keys and write keys at each level by using a public-key algorithm. Though this would eliminate the need for an assigned promoter, this would also more than double the number of keys stored by each member. In addition, as previously noted, the use of public-key encryption is not ideal since it is computationally slower.

The possession of multiple keys adds an aspect to the authentication problem. In non-differentially secure multicasting, there is no ordained way to authenticate a sender. All the receivers know is that the sender was authorized because the message was encrypted with the correct key. In differentially secure multicasting, an additional question arises regarding the security level at which the communication originated. Though an encrypted communication must come from an authorized sender, there is no designated way to determine at what level the sender is cleared. A member at level 4 can send a level 1 classified packet and the receivers of that packet would not automatically know what clearance level the sender has. However, all recipients of a level 2 packet would know that the originator is at least cleared as high as level 2. Thus, a packet at the highest level can only have come from one of the few members cleared at that level.

Also analogous to non-differentially secure multicasting, the members of a differentially secure multicast group do not necessarily know who within the authorized set of senders sent a packet. There is no prohibition against using an authentication scheme, such as those mentioned in Section 2.5, but there is no explicit prescription for one either. This allows for flexibility of choice when implementing differentially secure multicasting. One reason authentication might be used is to reestablish the "no write-down" rule of multilevel security. A sender who is forced to authenticate at a particular level could not evade the rule and compose a packet at a lower level.

The following approaches are assessed in terms of the *total tree cost*, the cost of sending one packet at each security level originating from the core node of the tree at that security level. The performance is measured with respect to the bandwidth consumed per packet transmission. The method by which this is measured accompanies the explanation of the approaches.

4.2 Naïve Approach

The Naive solution to differentially secure multicasting is to create separate multicast trees for each level of security, as shown in Figure 4. For the sake of illustration, only a two-level clearance is shown. Setting up regularly secure multicast trees is a well-known procedure. All group members at level 1 clearance are in the level 1 tree, all group members at a level 2 clearance are in the level 2 tree, etc. The obvious disadvantage is that assigning an address for each tree consumes more address space than a single tree. This becomes a multiplicative resource consumer if multiple differentially secure trees are instantiated.

The core may or may not be the same for each tree. If the core is not the same for each tree, there must be a way for lower-level packets to reach higher-level members. One way is to have all cores attach to the level 1 group, cores for groups higher than 2 attach to level 2, etc. However, this would create redundant packets on higher levels. The level 3 core would receive a level 1 message from the level 1 core and the same message from the level 2 core. The level 4 core would receive three copies of each level 1 packet and two copies of each level 2 packet. A better method is to have the cores of each tree become a member of the tree of the next lowest level. In this connection scheme, each level only receives one copy of each lower level's packets. Note that in Figure 4, node 2 sends a level 1 packet to node 1 which must then forward that to all level 2 members. To send a packet to node 8, the link between node 2 and node 1 is utilized. Thus, two packets are transmitted over the path between nodes 1 and 2 for each level 1 communication.

The total tree cost for the naïve approach is calculated in the following manner. Assuming all links to have equal weight, the number of links over which a packet must flow to reach all receivers is counted. In Figure 4, (where N designates node number and L designates security clearance) the cost to send a level 2 packet is calculated by adding the

links from nodes 1 to 6, 6 to 7, 1 to 2, and 2 to 8 for a cost of 4. The cost to send a level 1 packet is calculated by adding the links from node 2 to 1, 1 to 9, 2 to 3, 3 to 4, 4 to 5, plus the cost of sending a level 2 packet since the level 1 packet is forwarded to the level 2 tree by the core of level 2. The cost for sending a level 1 packet is 9. The total cost for the tree is calculated by summing the cost for all levels.

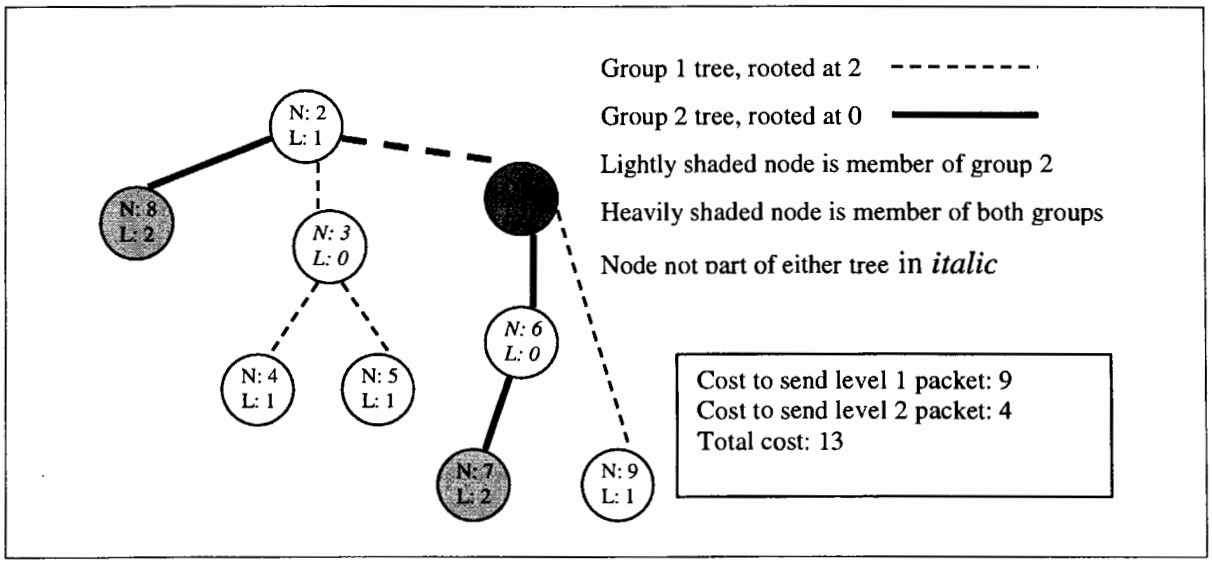

Figure 4. Naïve Approach

Having separate cores can make group communications more robust; if one core fails, the other levels can still communicate. However, this also means more machines are needed that have enough resources to manage this information. If the trees have the same core, there is an increased burden on this one core to manage multiple trees.

In either case, the core(s) for groups of level greater than 1 must be able to take a packet from a lower level, readdress it to the higher level(s) and forward the packet to these addresses. This readdressing requires that there be a label on the packet designating its security level. Though any level 1 receiver would automatically know to use the only key available, any level 2 or above receiver needs to choose which decryption key is needed to unlock the message. This label would only consume a few more bits per packet so it would consume trivial additional bandwidth. The router need not note the label since it can forward the packets appropriately based on the multicast address. The disadvantage of using such

labels is that an adversary can easily see it and use it to analyze traffic and well as know which packets are most desirable for cryptanalysis. An opponent would also know which machines communicated at the highest level and would be able to concentrate attacks on those machines in hopes of gathering more keys upon successful intrusion. Group members in any multicast scheme involving secure communications, differential or not, must practice best-known methods to prevent compromise of their keys.

4.3 Multiple Tree DiffSec Approach

The multiple tree differentially secure (DiffSec) solution, like the Naive scheme, creates separate multicast trees for each level of security, as shown in Figure 5. As in the Naïve scheme, the disadvantage is consumption of address space for each tree. The difference in this case is that all members are in the level 1 multicast tree, group members at clearance level 2 or above are in the level 2 multicast group, etc.

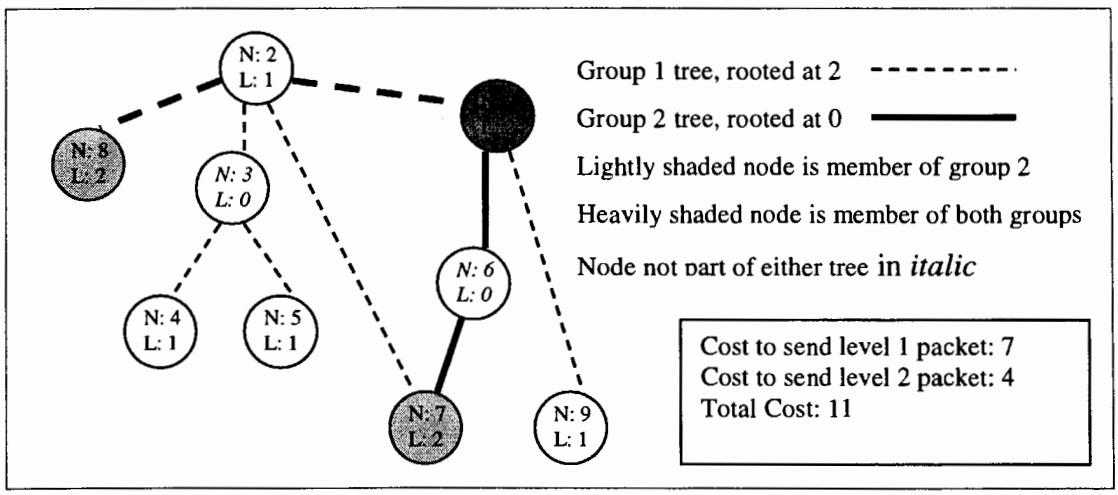

Figure 5. Multiple tree DiffSec example

The core may or may not be the same for each tree. As mentioned above, having separate cores allows for robustness but consumes management resources. In this case, the communication overhead for joins and leaves will be greatly increased since each joining and leaving member at level 2 or greater will have to contact multiple cores. Unlike the Naive approach, there is no need to label each packet with the appropriate security level; it is inherently known by the address of the group. The total tree cost is calculated in the same

manner as in the naïve approach. Note that the cost for node 7 to receive a level 2 packet from node 1 is 2 while the cost for node 7 to receive a level 1 packet from node 2 is 1.

4.4 Single DiffSec Tree Approach

The single tree DiffSec approach is to set up one multicast tree that has all levels in one tree. The setup time for this is a little longer since each router has to know the highest level of each link to a downstream node so that it can forward all packets at that level or below. This means that node 1 in Figure 5 must know that it should forward only packets of level 1 since the clearance level of node 1 is 2. This value is known as the maxchild and indicates the maximum-security clearance of all of its children. True, the router must store the extra information but in comparison to storing multiple address entries, as in the Naive scheme, this classification level information would require little space in memory. The increased setup time stems from the need to propagate this maxchild value up the tree.

In the worst-case situation, if the member who joins as a leaf node has the highest security clearance of its branch, this information would need to propagate up to the top of the tree. With a tree of height h, this means as many as h update maxchild messages could be sent per joining member. Updating the maxchild value when a member leaves generates more packets than the join in the worst-case. Using the instance of a leaf member with the highest clearance level on its branch, leaving the group would propagate to the immediate parent and the parent would have to check all of its children to determine which is now the maxchild. (The query is only sent to the immediate children and does not need to recurse down the tree since the values are refreshed for each join). This parent then sends the update maxchild message and its parent, the grandparent of the leaving member, repeats the query of children to determine if this new value is the max or not. The message continues to propagate until the new update maxchild message does not cause a change in the value of the receiver of the message. In the worst-case, this message propagates up to the core of the tree. For a d-ary tree of height h, the cost of updating the maxchild upon a leave could be as much as $O(d*h)$.

Not only must the router know the levels of the links, it must also know the classification of the packets. This can be accomplished using classification labels, as

mentioned in the Naive scheme. The same drawback of potential traffic analysis exists. Despite the additional bits needed for the label, the single DiffSec tree would actually conserve bandwidth. In the Naive approach, the separate trees may have links in common. There is the potential that a level one packet transmitted to all trees, would flow multiple times over the same link. With all the members in one tree, only one copy of a packet would be sent per link, instead of possible multiple copies as in the Naive approach.

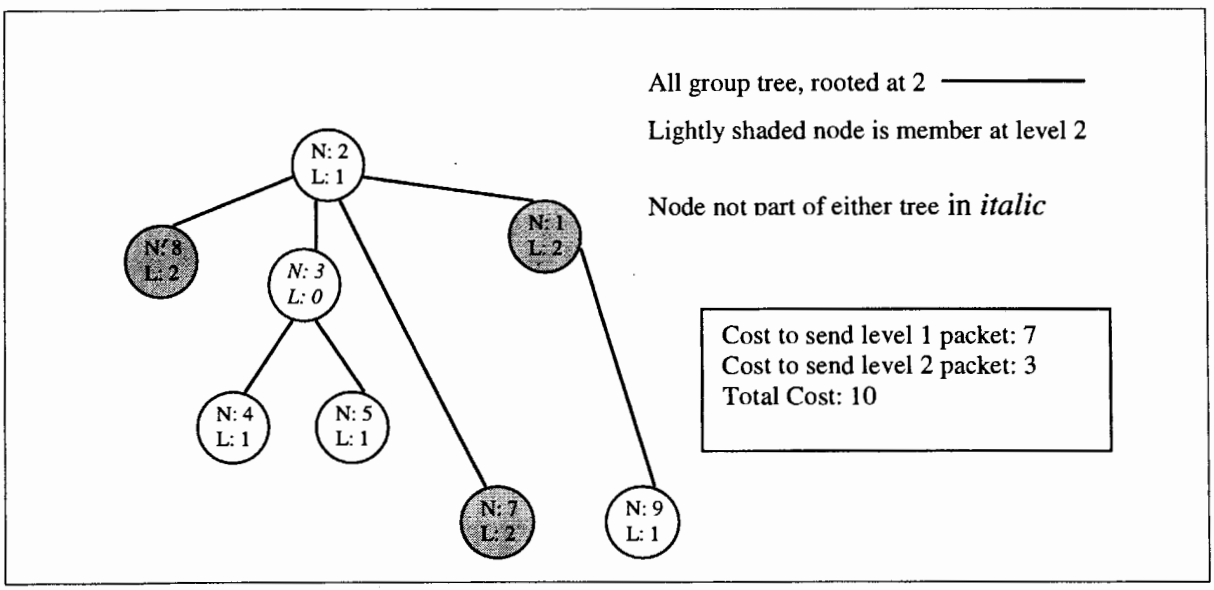

Figure 6. Single DiffSec tree example

4.5 Comparison

The naïve, multiple tree, and single DiffSec tree approaches are compared below in Table 2. The comparison is based on the metrics of scalability and link cost. Each approach's ability to scale relates to the ease of implementation, the allowance of simultaneous communication levels, and the amount of resource consumption. Ease of implementation is determined by the compared setup times and the number of controllers needed to facilitate the scheme. The Naïve and multiple tree approaches, which rely on separately creating trees, will have longer setups time than the single DiffSec tree, which will only take the same initialization time as the level 1 tree in the multiple tree approach. The Naïve and multiple tree approaches may have as many as one controller for each group. The

Table 2. Comparing Naïve, Multiple Tree, and Single DiffSec Tree Approaches

Category	Metric	Naïve	DiffSec	
			Multiple Tree	Single Tree
Ease of Implementation	Set up time	Multiple groups	Multiple groups	Single group
	Number of controllers	Up to one per group	Up to one per group	One
Simultaneous communications of levels		Allow	Allow	Allow
Resource Consumption	Address space	Multiple multicast addresses	Multiple multicast addresses	Single multicast address
	Routing Entries	Redundant entries for a single link	Redundant entries for a single link	One entry per link
	Maxchild update messages on join	None	None	Up to h
	Maxchild update messages on leave	None	None	Up to d*h
Link Cost per packet	Sparse groups*	Highest	Moderate	Lowest, if core at highest level
	Pervasive groups*	Highest	Moderate	Lowest
	Lightly connected network*	Highest	Moderate	Lowest, if core at highest level
	Densely connected network*	Highest	Moderate	Lowest, if core at highest level

(* from experimental data)

single DiffSec tree only has one controller. All schemes allow members at different levels to communicate simultaneously. The amount of resource consumption is evaluated in terms of the number of addresses consumed and the number of routing table entries needed. The naïve and multiple tree approaches will consume more multicast addresses (one for each tree) than the single tree approach. The Naïve and multiple tree solutions will also require more routing table entries because each multicast address will be maintained separately. Links for

higher level members may possibly be referenced multiple times. The total tree cost for the Naïve approach should always be the greatest since this case has the greatest overlap of communication links. The cost for multiple trees can be smaller than the cost for a single DiffSec tree for sparse groups but for pervasive groups, which have more path options to the core of the tree, the single DiffSec tree will cost less on average than a multiple tree DiffSec implementation. For a lightly connected network, one with has a smaller average degree of links per node, and a densely connected network, the multiple tree and single tree cases result in equal costs and the Naïve case has the greatest cost. The cost of the single tree case is improved when choosing the core to be at the highest level instead of the lowest level.

5. EXPERIMENTS

The experiments in this thesis involve performing cost analyses of the three arrangements of differentially secure multicast groups. The software program ns, created at the University of California, Berkeley, was used to construct networks and multicast groups with varying characteristics. For each randomly connected network and multicast group with given parameters, the three differentially secure schemes were constructed and the costs were calculated. The following chapter explains how the setup of the experiments and their results.

5.1 Experiment Setup

First, a network was constructed with a given number of nodes and a given number of links. For processing purposes, the number of nodes was limited to 64. The randomness in the connectivity was achieved using the random number generator available in ns. By seeding this function with predefined good seeds, ns can create a list of random node connectivities that is statistically independent. The number of links ranged from 96 to 256 in steps of 16. Specifying the number of links controls the density of the network and the number of available paths for group communications. The random nature of assigning these links caused some redundant connections. The average degree of each node ranged from 2.4 (when 96 links were created) to 5.8 (when 256 links were created).

Once the network was constructed, the ns random number generator was used to randomly and uniformly assign levels to a given number of receivers. The number of receivers was varied from 12 to 36 in steps of 4 since the experiments in this thesis are limited to groups with 4 levels. Electing receivers among nodes of the network was randomized so that any node may be picked to be a receiver. Once receivers were elected, that group was used between experiments of the same number of receivers. In one case, a uniform distribution was enforced so that each security level had an equal number of receivers. In the second case, the receivers were distributed nonuniformly so that approximately 40% of the receivers were at level 1, 25% were at level 2, 20% were at level 3, and 15% were at level 4. The nonuniform distribution simulates the case of a group with a

few managers and many employees. To illustrate the randomness of the configuration in a manageable fashion, Figure 7 shows a randomly generated 10-node network with 20 links and 6 receivers.

Once the network was established and the receivers defined, the following algorithms in Figures 8 through 11 were used to construct differentially secure groups using each scheme and count the cost of that scheme. Each multicast group had a cost variable associated with it. The total cost is calculated by counting the cost for each group if a packet at each level was transmitted and adding all the group costs together. As each node joined, that cost variable for the group was updated to reflect the number of links traveled for a communication packet to reach all members. In the Naïve scheme, the total cost is found by adding the cost for each group and multiplying it by its level to achieve the actual cost for the group. A level 3 tree that has a cost of 4 links would have an overall cost of 3*4=12 since the 4 links must propagate level 1, level 2, and level 3 packets. In the multiple tree scheme, the groups includes everyone at or below that level so there is no need to multiply the group cost and the level of the group. Simply total the group costs for all levels. In the single

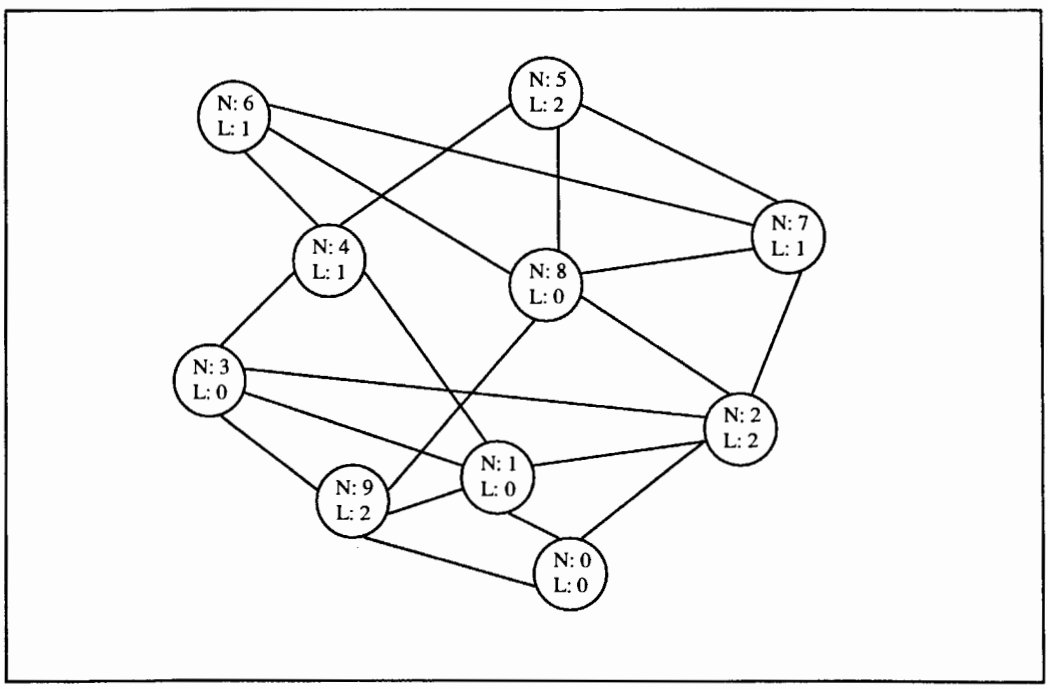

Figure 7. Randomly generated 10 node network with 20 links and 2 levels [for illustration only]

DiffSec tree case, a value must be kept up-to-date at each node that indicates the highest clearance level of that node's children. If the node is a leaf node, this variable, maxchild, will be equal to the level of the node. A single DiffSec tree, created from the random network shown in Figure 7, is shown in Figure 12 with the maxchild variable of each node indicated.

```
ConstructNaiveDiffSec{
      For level from 1 to 4
            Assign lowest node number at that level to be the
              core for group #level
      For all nodes in the tree
            If node has a non-zero level
                  join node to the group at its level
      For all cores of groups greater than level 1
            Join core to core of the group of the next lowest
              level
      For levels 1 to 4
            Set total cost to total cost +
                              (cost of group at level * level)
}
```

Figure 8. Evaluating Naïve DiffSec scheme

```
ConstructMTreeDiffSec{
      For level 1 to 4
            Assign lowest node number at that level to be the
              core for group level
      For all nodes in the tree
            If node has a non-zero level
                  Join node to all groups from 1 to level
      For levels 1 to 4
            Set total cost to total cost + cost of group at
              level
}
```

Figure 9. Evaluating multiple tree DiffSec cost

```
ConstructSTreeDiffSec{
      For level 1
            Assign lowest node number at level 1 to be the
              core for the group
      For all nodes in the tree
            If node has a non-zero level
                  Join node to the group
      For all nodes except the core
            Set cost to cost + max(maxchild(node),level(node))
}
```

Figure 10. Evaluating single DiffSec tree cost

```
maxchild(node) {
      If level of node is non-zero and node is a leaf node
            maxchild = level of node
      else if level of node is non-zero
            maxchild = max(maxchild values of all children of
                           node)
}
```

Figure 11. Updating maxchild value for single DiffSec tree

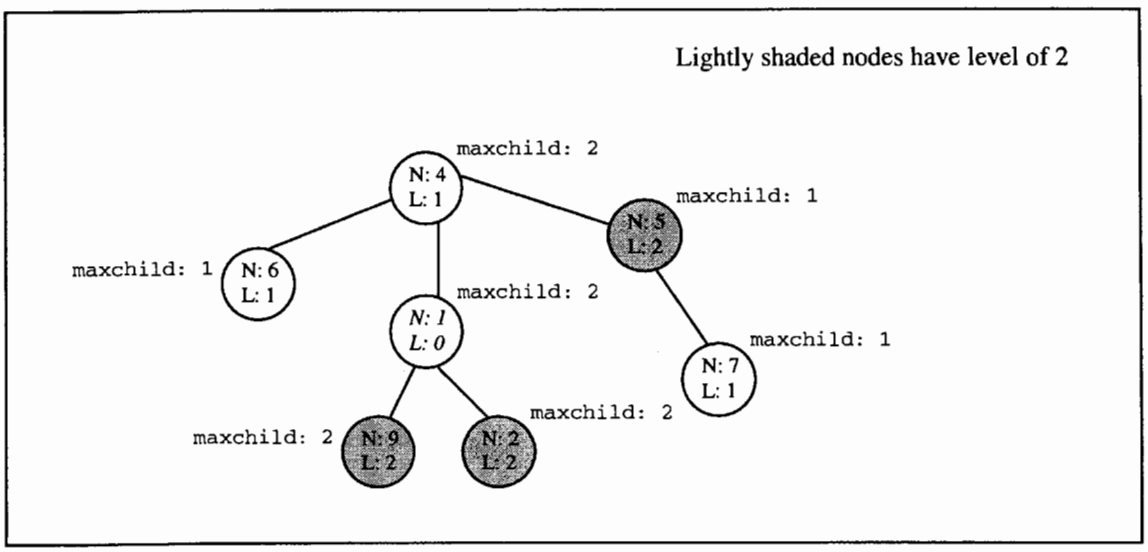

Figure 12. Single DiffSec tree example

5.2 Experimental Results

Each simulation run had 60 trials, where a trial means creating a network with a given number of links, a given number of receivers and counting the cost of each of the three schemes. The single tree scheme was constructed in two different manners, one in which the core was at level 1 and a second in which the core was at level 4. On average the Naïve scheme has the highest cost, the single DiffSec scheme with a level 1 core the next highest, the multiple tree DiffSec scheme the third highest, and the single DiffSec scheme with a level 4 core the lowest cost, as seen in Figures 13, 14, 15 and 16. In Figures 13 and 14, the number of receivers is varied from 16 to 36 in steps of 4. When 50% of the nodes (32 nodes) in the network are part of the differential security group, the single DiffSec tree scheme with a level 1 core begins to exhibit a cost-savings over the multiple tree scheme. The cost difference between the multiple tree scheme and the single tree scheme is within the range of the number of levels in the group. By simply moving the single DiffSec tree core from group 1 to group 4, the cost difference between the multiple tree and single DiffSec tree schemes may be eliminated or even improved, as seen in the figures. With 95% certainty, the true means for the schemes in Figures 13 and 15 are within the range ±4.561, ±2.463, ±2.947, and ±2.882 for the Naïve, multiple tree, single tree with level 1 core, and single tree with level 4 core respectively. The true means for the schemes in Figures 14 and 16 are have a 95% probability of being within the range ±4.057, ±2.259, ±2.937, and ±2.502 for the Naïve, multiple tree, single tree with level 1 core, and single tree with level 4 core respectively.

In Figures 15 and 16, the number of links is varied from 96 to 256 in steps of 16. As the number of links is increased, the cost of the trees decrease since there are more paths available to the core so the hop length between each member and the core is less. As above, the multiple tree scheme and the single DiffSec tree have nearly equal costs while the naïve scheme the highest cost. The difference between the multiple tree and single DiffSec tree schemes is always less than four. Changing the core to be at level 4 reduces the cost of the tree.

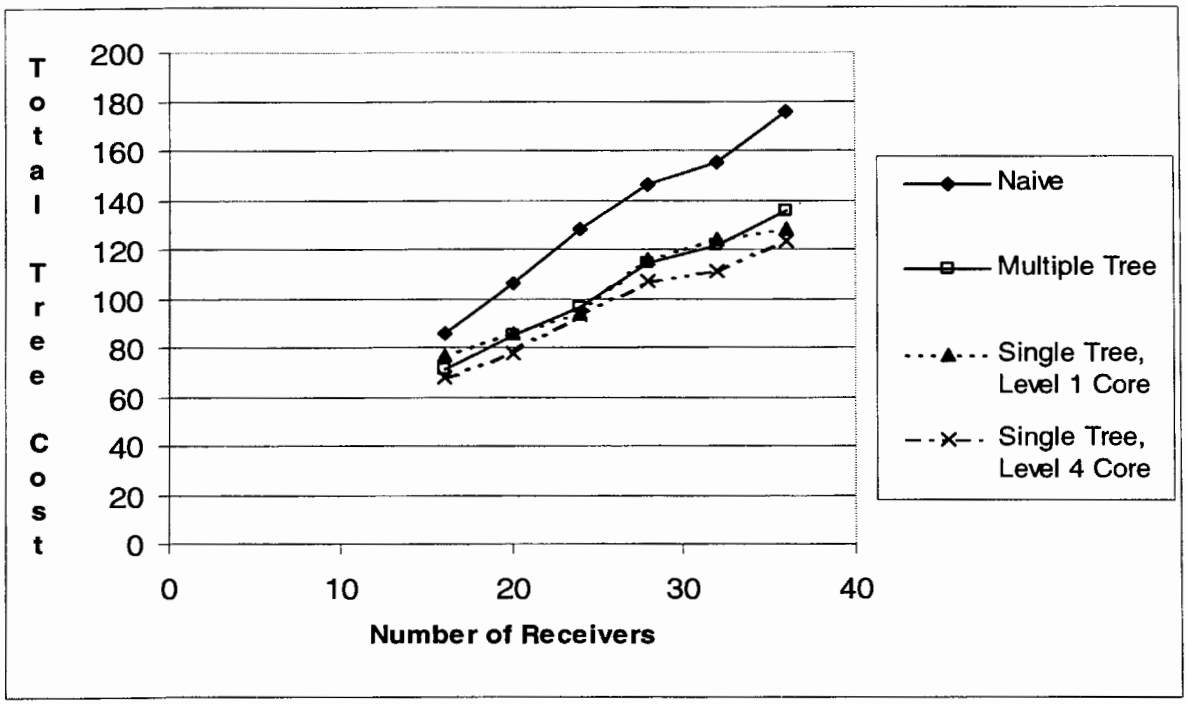

Figure 13. Effect of number of receivers on tree cost, uniform case

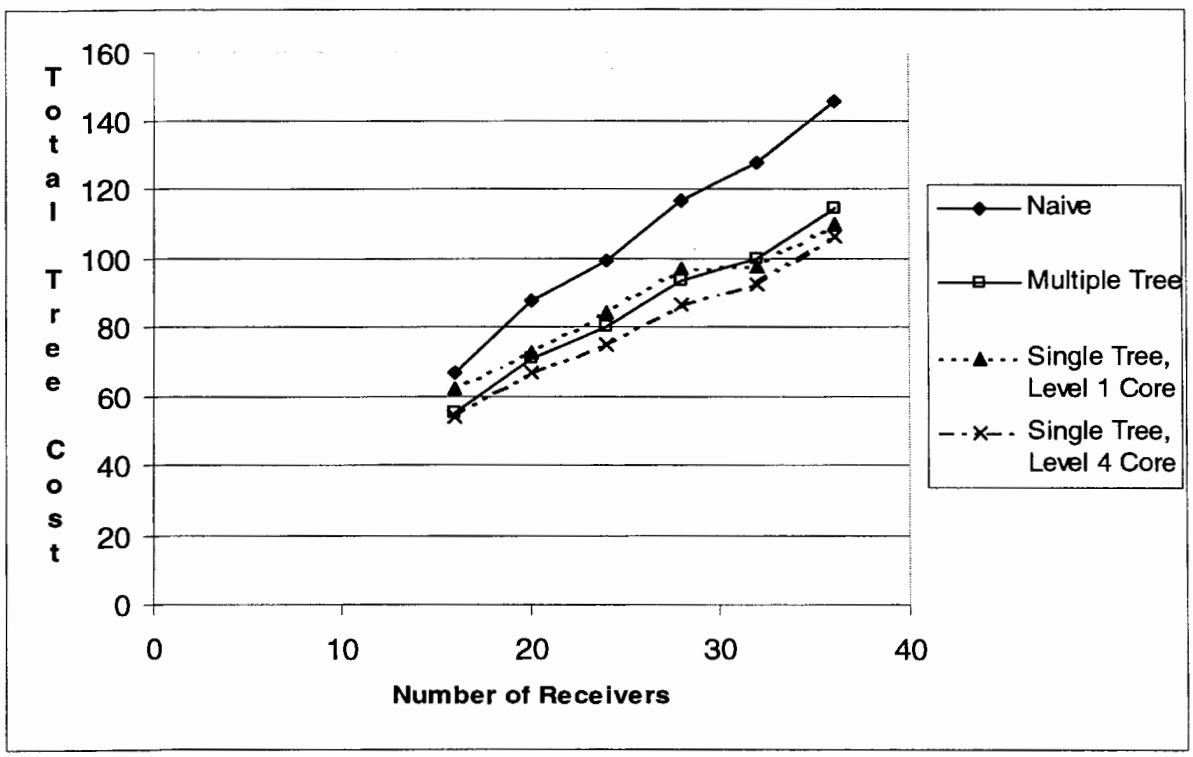

Figure 14. Effect of number of receivers on tree cost, nonuniform case.

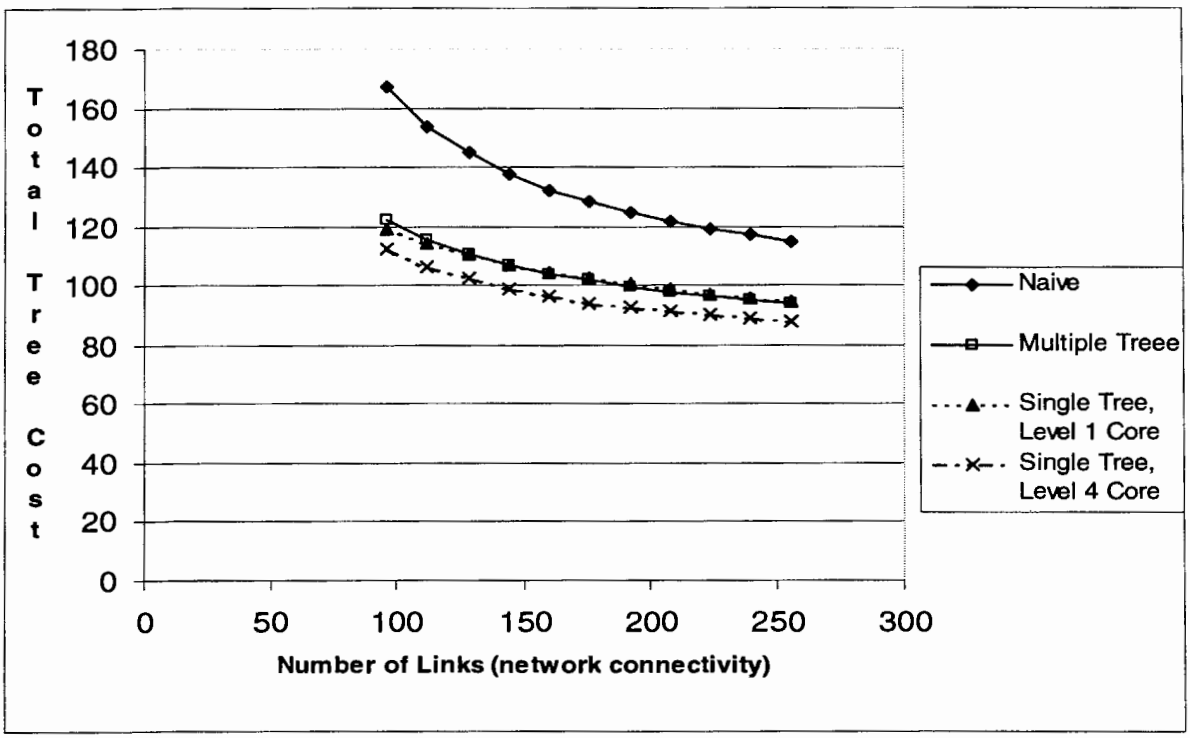

Figure 15. Effect of network connectivity on tree cost, uniform case.

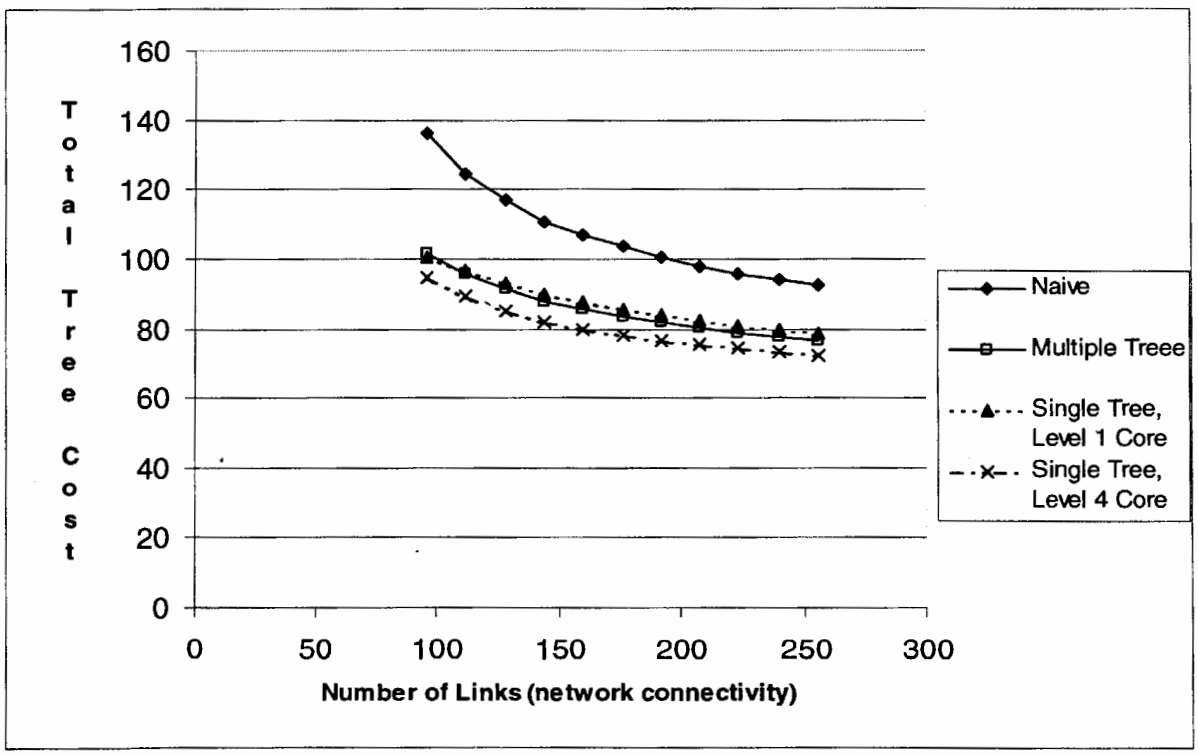

Figure 16. Effect of network connectivity on tree cost, nonuniform case.

6. CONCLUSIONS

With the explosion of the use of electronic means for communication there has been a corresponding increasing desire to protect and ensure information as it is transmitted. The means of information assurance involves the use of various cryptographic systems. At the same time as the spreading employment of encryption to secure messages is the burgeoning of multicasting to support group communication applications. The challenge faced is how to combine the two fields and provide secure multicast communications.

Secure multicasting has been studied from the aspects of group architecture, group key management and sender authentication. There are multiple methods to implement each of these topics. The characteristics of the multicast group determine which of these methods or combination of these methods will be most beneficial for a particular situation. One area not explored in secure multicasting is the implementation of multilevel secure encrypted communications. Multilevel security has been considered in the sender authentication issue [WL99] but not in terms of securing communications at multiple sensitivity levels within a group. Project groups, in both the military and business communities, have multilevel security attributes and can benefit from the development of a multilevel secure group communication scheme.

The concept proposed in this thesis is that of differentially secure multicasting. Based from concepts in the Bell-La Padula multilevel security model, differentially secure multicasting relaxes some of the rules so as not to prevent necessary communications between group members at differing security levels. The differential security model can be integrated into a multicast setting in three ways. These approaches were compared using simulated networks and with varying characteristics.

In all cases, the Naïve set up scheme proved to be the worst option. Maintaining separate trees for each level of communication increases the number of packets because some links between nodes exist in multiple groups. A group, not knowing about the membership or routing paths of the other groups, may use part of the same path another group used to send same packet to a different member. The drawbacks of the naïve scheme overshadow the benefit of easy set up.

The multiple tree and single DiffSec tree approaches have nearly equal link costs. In the single DiffSec tree scheme, selecting a core at the highest level results in a cost savings that brings the single DiffSec tree cost in-line with or less than the multiple tree scheme. The equality of costs of these schemes means that other factors must be considered when choosing which one to implement. The benefits of the multiple tree scheme include graceful failure, since the cores of the trees can be separate from one another, and the ability to be implemented using existing technology since routers send packets based on the address of the group without regard to security level labels on packets. The benefits of the single DiffSec tree scheme include a conservation of multicast addresses and a reduction of routing entries needed to send packets to each member. For a pervasive group where group members consist of 50% or more of the network, the single tree scheme is the best option since it has the lowest cost.

There is much future work to be done for the single DiffSec tree scheme to be deployed. The protocols for propagating and updating the maxchild values at each node need to be defined. The optimization algorithms used to minimize the cost of communications by carefully choosing a core would need to take into account the level of the core when making decisions. The packet format must be defined to include the security label. With the growing use of secure multicast communications and the demonstrated benefits of the single DiffSec tree, implementations of this scheme would be desirable and useful to the business and military community alike.

REFERENCES

[Be00] Clifford Bergman. Lectures in cryptography. Iowa State University, Spring 2000.

[BFC93] T. Ballardie, P. Francis, and J. Crowcroft. "Core-based trees (CBT): An architecture for scalable inter-domain multicast routing." In *Proc. ACM SIGCOMM*, pp. 85-95, 1993.

[BL73] D. Bell, and L. La Padula. "Secure computer systems: Mathematical foundations and model". *MITRE Report,* M74-244, MTR 2547 v2, November 1973.

[CD85] D.R. Cheriton and S.E. Deering. "Host Groups: A Multicast Extension for Datagram Internetworks". In *9th Data Communication Symposium*, IEEE Computer Society and ACM SIGCOMM, September 1985.

[CG98] D. Catalano and R. Gennaro. "New efficient and secure protocols for verifiable signature sharing and other applications." In *Proc. CYRPTO'98*, LNCS vol. 1642, Springer-Verlag, pp.105-20, 1998.

[CGIMNP99] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and B. Pinkas. "Multicast security: A taxonomy and some efficient constructions". In *Proc. IEEE Infocom*, March 1999.

[Ch98] I. Chang, et al. "A toolkit for secure multicast services over the Internet." IBM Technical Report, 1998.

[D88] S. Deering. "Multicast routing in internetworks and extended LANs." In *SIGCOMM '88*, Stanford, CA, pp. 55-64, August 1988.

[DH76] W. Diffie and M. E. Hellman. "New directions in cryptography". *IEE Transactions on Information Theory*, 22, pp. 644-54, (1976).

[DM78] Y. K. Dalal and R. M. Metcalfe. "Reverse path forwarding of broadcast packets." *Communications of the ACM*, vol. 21, no. 12, pp. 1040-8, December 1978.

[GS94] L. Gong and N. Shacham. "Elements of trusted multicasting". In *Proceedings: 1994 International Conference on Network Protocols*, IEEE Computer Society Press, October 1994.

[HCD00] T. Hardjono, B. Cain, and N. Dorawswamy. "A framework for group key management for multicast security". IETF Internet draft, August 2000. draft-ietf-ipsec-gkmframework-03.txt

[KBC97] H. Krawczyk, M. Bellare, and R. Canetti. "HMAC: Keyed-hashing for message authentication". RFC 2104, February 1997.

[M97] S. Mittra. "Iolus: A framework for scalable secure multicasting". In *Proc. ACM SIGCOMM,* 1997.

[MRR99] M. J. Moyer, J. R. Rao, and P. Rohatgi. "A survey of security issues in multicast communications". *IEEE Network*, pp. 12-23, Nov/Dec 1999.

[NBS77] Data Encryption Standard (DES). National Bureau of Standards FIPS publication 46, 1977.

[Pf97] Charles P. Pfleeger. *Security in Computing*. New Jersey: Prentice-Hall, 1997.

[PMZ99] M. Peyravian, S.M. Matyas, and N. Zunic. "Decentralized group key management for secure multicast communications". *Computer Communications* 22, pp. 1183-7, 1999.

[QA99] B. Quinn and K. Almeroth. "IP multicast applications: Challenges and solutions". IETF Internet-draft, June 1999. draft-ietf-mboned-mcast-apps-01.txt.

[Ra00] M. Ramalho. "Intra and inter-domain multicast routing protocols: A survey and a taxonomy". *IEEE Communications Surveys & Tutorials*, vol. 3, no. 1, pp. 2-25, First Quarter 2000.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. "A method for obtaining digital signatures and public key cryptosystems." *Communications of the ACM*, 21, pp. 120-6, 1978.

[Sc94] B. Schneier. *Applied Cryptography*. New York: John Wiley & Sons, Inc., 1994.

[SGLA99] C. Shields and J.J. Garcia-Luna-Aceves. "KHIP -- A scalable protocol for secure multicast routing." *ACM SIGCOMM*, 1999.

[SM00] L. Sahasrabuddhe and B. Mukherjee. "Multicast routing algorithms and protocols: A tutorial". *IEEE Network*, pp. 90-102, January/February 2000.

[St99] William Stallings. *Cryptography and Network Security: Principles and Practice, 2nd ed.* New Jersey: Prentice-Hall, Chapter 9, pp. 271-97,
1999.

[St95] Douglas R. Stinson. *Cryptography: Theory and Practice*. Boca Raton, Florida: CRC Press LLC, 1995.

[WGL00] C. K. Wong, M. Gouda, and S. S. Lam. "Secure group communications using key graphs." *IEEE/ACM Transactions on Networking*, vol. 8, no. 1, pp. 16-30, February 2000.

[WL99] C. Wong and S. Lam. "Digital signatures for flows and multicasts." *IEEE/ACM Transactions on Networking*, vol. 7, no. 4, August 1999.

[YFM98] Y. Huang, E. Fleury, and P. K. McKinley. "LCM: A multicast core management protocol for link-state routing networks." *ICC*, pp. 1197-201, 1998.

ACKNOWLEDGEMENTS

First, I must thank my major professor, Dr. Manimaran Govindarasu. Without your willingness to accept me as a student and to guide me through this process, I would have been lost. I also want to thank Dr. Jim Davis for your encouragement throughout my school career at Iowa State and especially for your counseling during the thesis process. Thank you Anirban Chakrabarti for your lengthy hours tutoring me in the writing of TCL and the use of ns. Thank you Aaron Striegel for your tutelage on multicasting and your editing wisdom. Thank you to Tina Chang and her husband, the proprietors of Chinese Home Style Cooking. Your delicious food saved me time and gave me the sustenance to continue working long hours. I do not know how to express thanks enough to my siblings, my parents, and my friends for all of the love and emotional support you have given to me as I sought my degree.